~~Attorney Docket: SAP.203.17 US~~

Appendix B: Marked-up Version of Substitute Specification

SAP AG,
Dietmar-Hopp-Allee 16, 69190 Walldorf, Germany

A method of assigning objects to processing units

Description

Related Applications

This application is a national stage filing under 35 U.S.C. § 371 of International Application No. PCT/EP2004/0008103, filed on July 20, 2004, which published in the English language and claims the benefit of priority to European Application No. 03018101.0, filed August 8, 2003.

Field of the invention

The present invention generally relates to the field of data processing~~, and more~~. More particularly and without limitation, the invention relates to methods and systems related to object size balancing in a multi-computing environment.

Background ~~and prior art~~

Various multi-computing architectures are known ~~from the prior art~~ where a plurality of processing units [[is]] are coupled to form a cluster. Such architectures are used in parallel processing and also in the emerging field of blade computing.

Blade computing relies on blade servers, which are modular, single-board computers. An overview of blade computing is given in "Architectures and Infrastructure for Blade Computing", September 2002, Sun ~~microsystems~~ Microsystems and "THE NEXT WAVE: BLADE SERVER COMPUTING", Sun Microsystems ~~(www.sun.com/servers/entry/blade)~~.

A content load balancing blade is commercially available from Sun ~~microsystems~~ Microsystems, for example, the Sun Fire TM B10n ~~("Sun Fire TM B10n)~~. This blade provides traffic and content management functionalities. Content load balancing is achieved based on URLs, CGI scripts and ~~cookies; server~~ cookies. Server load balancing is achieved based on server loads, response times, and weighted round-robin algorithms.

US patent application no. 20030105903 shows a web edge server, which comprises a number of blade servers. A switch and an information distribution module are provided for the purpose of balancing. The information distribution module receives an information message, performs processing on the message to determine a destination, and forwards a message toward the determined destination via an internal communications network.

Summary of the invention

[[The]] Methods and systems consistent with the present invention ~~provides~~ are provided for a method of assigning objects to processing units of a cluster of processing units. Each one of the processing units has a certain storage capacity. For the purpose of balancing the sizes of objects of the individual processing units, a given number of objects needs to be distributed. This is accomplished by sorting of the objects by size, which provides a sequence of objects. This sequence is used for assigning of objects to processing units.

[[The]] In accordance with one embodiment of the invention, the procedure for assigning of objects to a processing unit, starts with the largest object of the sequence and continues until the remaining storage capacity of the processing unit is below the size of the smallest remaining object of the sequence. When this condition is fulfilled, the procedure is carried out again for the next processing unit, whereby the objects which have

been previously assigned, are deleted from the sequence. This way a minimum number of processing units, which are required for handling a given set of objects can be determined.

In accordance with ~~a preferred~~ another embodiment of the invention, each processing unit is a single-board computer that has a bus interface to a bus system that couples a plurality of the single-board computers. Each of the single-board computers ~~has its~~ may include private processing and data storage resources. Data processing tasks or subtasks of a complex data processing task are assigned to the single-board computers by a control unit. The control unit can be a separate hardware unit or a software process that runs on one of the single-board computers. An example of such a distributed data processing system is a cluster of blades.

In accordance with ~~a preferred~~ another embodiment of the invention the remaining storage capacity of a processing unit is determined by the difference between the storage capacity of the unit and the aggregated size of objects, which have been assigned to the processing unit. On the basis of this definition of the remaining storage capacity, the minimum number of processing units is determined.

In accordance with a further ~~preferred~~ embodiment of the invention, the object size balancing procedure is performed again in order to further improve the object size balancing. For this purpose the largest gap between the aggregated sizes of objects being assigned to one of the processing units and the maximum storage capacity is determined.

This gap is divided by the minimum number of processing units and the result of the division is subtracted from the maximum storage capacity to provide a threshold level. When the procedure for assigning the objects to the processing units is performed again, the definition of the remaining storage capacity is the difference between the aggregated size of the objects being assigned to the processing unit and the threshold level. As a result of the renewed performance of the assignment procedure, the gap can be substantially reduced.

In accordance with a further ~~preferred~~ embodiment of the invention, the theoretical storage capacity limit for a perfectly evenly distributed load is used as a threshold. This

threshold is obtained by calculating the difference between the total of the storage capacities of the processing units and the total of the sizes of the objects and dividing the difference by the minimum number of processing units. The result of the division is subtracted from the storage capacity, which provides the theoretical limit.

The assignment procedure is performed again, whereby the remaining storage capacity is defined as the difference between the aggregated size of the objects of a processing unit and the threshold. Typically the storage capacity of the last processing unit of the minimum number of processing units, to which the objects are assigned in the procedure, will not be sufficient to accommodate all of the remaining objects of the sequence.

In this case one [[ore]] or more iterations are performed. For one iteration the excess amount of memory is divided by the minimum number of processing units. The result of the division is added to the threshold and the assignment procedure is performed again. This process continues until the storage capacity of the last processing unit, to which the remaining objects of the sequence are assigned in the procedure, is sufficient to accommodate all these objects. This way, the object size balancing [[is]] may be further improved.

In accordance with a further ~~preferred~~ embodiment of the invention, the threshold for performing the assignment procedure is varied between the theoretical limit and the storage capacity. For each value of the threshold, a new assignment procedure is performed. For each of the assignments of objects to processing units, a statistical measure is calculated. This statistical measure is a basis to select one of the assignments for optimal object size balancing.

In accordance with a further ~~preferred~~ embodiment of the invention the standard deviation or variance of the sum of the object sizes assigned to a processing unit is used as a statistical measure. The standard deviations obtained for the processing units as a result of the assignment procedure are stored as an overall quality measure of the assignment. The assignment having the lowest overall quality measure is selected.

In accordance with a further ~~preferred~~ embodiment of the invention, each one of the processing units is a blade or a blade server. One of the blades can have a program,

which implements the principles of the present invention, in order to perform object size balancing. This way the number of swap-operations between the blades can be minimized.

In accordance with a further ~~preferred~~ embodiment of the invention the principles of the invention are implemented in an application program running on a personal computer. The application program is provided with a list of objects and the estimated sizes of the objects, which needs to be handled by the cluster of processing units. On the basis of the object sizes, the minimum number of processing units which are required for the processing can be determined. This information can form the basis for a corresponding investment decision of a customer.

It is to be noted that the present invention is not restricted to a particular type of objects. For example, data objects such as tables, arrays, lists, and trees are distributed to processing units, e.g. blades, in accordance with the principles of the present invention. For example, each one of the processing units runs a data processing task to which the respective objects are assigned.

Brief description of the drawings

In the following, preferred embodiments of the invention will be described in greater detail by making reference to the drawings in which:

Figure 1 is a schematic block diagram of [[a]] an exemplary modular computer system, having a cluster of blades, consistent with an embodiment of the present invention;

Figure 2 is ~~illustrative of~~ a flow diagram for an exemplary method for assigning of objects to blades and for determining the minimum number of blades, consistent with an embodiment of the present invention;

Figure 3 is an example [[for]] of tables, which need to be assigned to blades, consistent with an embodiment of the present invention;

Figure 4 ~~shows~~ illustrates the result of [[a]] an exemplary sorting operation, consistent with an embodiment of the present invention;

Figure 5 ~~shows a~~ illustrates an exemplary first step of assigning a table to a first one of the blades, consistent with an embodiment of the present invention;

Figure 6 ~~shows a~~ illustrates an exemplary second step for assigning a table to the first blade, consistent with an embodiment of the present invention;

Figure 7 shows the first assignment of a table to a second blade, consistent with an embodiment of the present invention;

Figure 8 shows a second assignment of a table to the second blade, consistent with an embodiment of the present invention;

Figure 9 shows the assignment of three further tables to the second blade, consistent with an embodiment of the present invention;

Figure 10 shows the resulting assignment of tables to blades as a result of the assignment procedure, consistent with an embodiment of the present invention;

Figure 11 ~~is illustrative of a preferred embodiment of the invention,~~ illustrates an example where the procedure of ~~figure~~ Figure 2 is performed again with a lower threshold, consistent with an embodiment of the present invention;

Figure 12 ~~is illustrative of~~ illustrates the lower threshold, consistent with an embodiment of the present invention;

Figure 13 ~~is illustrative of~~ illustrates the result of the renewed performance of the procedure of ~~figure~~ Figure 2 with the lower threshold, consistent with an embodiment of the present invention;

Figure 14 ~~is illustrative of a preferred embodiment of a~~ illustrates an exemplary method of the invention where the threshold is varied iteratively, consistent with an embodiment of the present invention;

Figure 15 ~~is illustrative of~~ illustrates the starting point of the iteration, consistent with an embodiment of the present invention;

Figure 16 ~~shows~~ illustrates the result of the first iteration, consistent with an embodiment of the present invention;

Figure 17 ~~shows~~ illustrates the resulting assignment of objects to the minimum number of blades after completion of the procedure of ~~figure~~ Figure 14, consistent with an embodiment of the present invention;

Figure 18 ~~is illustrative of~~ illustrates a further ~~preferred embodiment of the invention,~~ example where the threshold is varied in predetermined steps, consistent with an embodiment of the present invention;

Figure 19 ~~is illustrative of~~ illustrates the discrete continuum in which the threshold is varied and the result of the assignment procedure, consistent with an embodiment of the present invention;

Figure 20 ~~is illustrative of a~~ illustrates an exemplary computer system performing the assignment of objects to blades.

Detailed description

Figure 1 is a schematic block diagram of an exemplary modular computer system, having a cluster of blades, consistent with an embodiment of the present invention. Figure 1 shows cluster 100 of blades B_1, B_2, B_3, ..., B_N. Each one of the blades has processor 102 and memory 104. In the example considered here, all memories 104 have the same storage capacity. The blades are coupled by a network 106, such as a

bus system. The number N of blades of cluster 100 needs to be chosen, such that a given number of M objects of varying sizes can be handled.

For example, cluster 100 implements a ~~so-called~~ search engine. In this instance identical search processors run on each one of the blades. The assignment of data objects, such as index tables, to blades can be stored in a dispatcher unit (not shown in the drawing) of cluster 100. ~~This way~~ As a result, data objects are assigned to blades and data processing tasks running on the blades.

Figure 2 is a flow diagram for an exemplary method of assigning of objects to blades and for determining the minimum number of blades, consistent with an embodiment of the present invention. Figure 2 shows the corresponding procedure for assigning the objects to blades and thereby determine the minimum value for N.

In step 200, a sorting operation is performed in order to sort the M objects by size. The corresponding object sequence is provided in step 202. In step 204 the index i for the blades is ~~initialised~~ initialized to one.

In step 206 processing of the object sequence starts in the order starting with the largest object of the sequence. The first object of the sequence, which by definition is the largest object of the sequence, is assigned to blade B_1 in step 206. In step 208 the first object which has been assigned to blade B_1 is deleted from the sequence.

In step 210 the size of the objects, which have been already assigned, to blade B_1 is added up and a gap G between the aggregated object size and a threshold is calculated. When the assignment procedure of Fig. 2 is carried out for the first time, the threshold is the storage capacity of one of the blades.

In step 212 it is determined whether there remains an object in the sequence, which fits into the gap G. If this is the case, the largest of these objects is assigned to the blade B_1 in step 214 and deleted from the sequence before the control goes back to step 210.

If there is no such object which fits into the gap G, step 218 is carried out. In step 218 it is determined whether all objects have already been assigned to blades. In other words,

in step 218 it is checked whether the sequence is empty. If this is not the case the index i is incremented in step 220 and the control goes back to step 206 to assign remaining objects of the sequence of the next blade B_2.

If the contrary is the case, the index i equals the minimum number N of blades which are required to handle the M objects. This number is outputted in step 222. The minimum number N of blades can be a basis for an investment decision for purchasing of a corresponding number of blades. Further, the assignment of objects to blades is outputted in step 224 in order to visualize the quality of the object size balancing.

Figure 3 ~~shows an example~~ is an example of tables, which need to be assigned to blades, consistent with an embodiment of the present invention. In the example considered here, the objects are a number of twenty different tables having various sizes between 50 MB and 3566 MB as indicated in ~~figure~~ Figure 3. For example, table 1 has a size of 3250 MB, table 2 has 250 MB, table 3 has 750 MB, etc. The table sizes can be actual table sizes or average table sizes which have been obtained by monitoring a real life data processing system. Alternatively, the table sizes are estimates for the purpose of planning cluster 100.

Figure 4 shows the result of the sorting operation performed on the tables 1 to 20 of ~~figure~~ Figure 3 (cf. step 202 of ~~figure~~ Figure 2), consistent with an embodiment of the present invention.

Figure 5 illustrates the assignment of the first object of the sequence, i.e., the largest table 20 to blade B_1, consistent with an embodiment of the present invention. In the example considered here, each blade has a storage capacity of 4 GB = 4096 MB of main memory. Table 20 has a size of 2566 MB, which leaves a gap G of 530 MB of remaining storage capacity (cf. step 210 of ~~figure~~ Figure 2).

~~Next it~~ It is then determined whether there is a next object in the sequence which fits into the gap G. Table 12, which has a size of 520 MB, is the largest table which fits into the gap G. This table 12 is thus also assigned to blade 1. The aggregated size of the objects assigned to blade 1, i.e., table 20 and table 12, is 4068 MB, which leaves a gap

G of 10 MB. This gap G of 10 MB is too small to accommodate even the smallest remaining object of the sequence of tables.

~~As there remain~~ Because tables remain in the sequence which have not yet been assigned to a blade, the index i is incremented and the assignment procedure goes to the next blade B_2 (cf. steps 218 and 220 of ~~figure~~ Figure 2). With respect to blade B_2 the above-explained procedure is carried out again on the basis of the unassigned tables, which remain in the sequence.

~~This way~~ As a result, the largest remaining table of the sequence, i.e., table 15, is assigned to blade B_2 which leaves a gap G of 596 MB. The gap G is filled with tables 6, 2, 13 and 14 as illustrated in ~~figures~~ Figures 7 and 8. The resulting assignment of tables to blade B_2 is shown in ~~figure~~ Figure 9.

The aggregated size of the tables, which have been assigned to blade B2, i.e., tables 15, 6, 2, 13 and 14, leave a gap G of 76 MB which is not enough to accommodate the smallest unassigned table, i.e., table 11, of the sequence. Thus, the index i is incremented and the assignment procedure is continued for the next blade B_3. This process goes on until all tables of the sequence have been assigned to one blade B_i. The result of the assignments of tables to blades is illustrated in ~~figure~~ Figure 10.

In addition to the assignment of tables to blades, ~~this way~~ the minimum number N of blades, which are required for handling of the given number of tables (cf. ~~figure~~ Figure 3), is also obtained. In the example considered here, the resulting assignment of tables to the N = 8 blades leaves a gap G of 2196 MB on blade 8. In order to further improve the object size balancing the method of ~~figure~~ Figure 11 is carried out.

In step 1100 the largest gap G is determined. In the example shown in ~~figure~~ Figure 10, this is the gap G of blade B_8. The other blades B_1 to B_7 have smaller gaps between the aggregated size of the tables assigned to the corresponding blade and the storage capacity of 4 GB.

In step 1102 the gap G determined in step 1100 is divided by the number N of blades. In the example of ~~figure~~ Figure 10, this means that G = 2196 MB is divided by N = 8 in

order to obtain the value of Delta 1 = 275 MB. In step 1104 a threshold is calculated by subtracting Delta 1 from the storage capacity, i.e., threshold = 4096 MB - 275 MB = 3821 MB.

With the threshold calculated in step 1104, the method of ~~figure~~ Figure 2 is performed again in step 1106. The resulting assignment of the objects to the blades is more evenly distributed due to the lowering of the threshold. This is illustrated by way of example in ~~figures~~ Figures 12 and 13 for the example of ~~figure~~ Figure 10.

Figure 12 shows the threshold T, which has been calculated in step 1104, consistent with an embodiment of the present invention. With the lowered threshold T, the assignment procedure of ~~figure~~ Figure 2 is restarted from the beginning whereby steps 200 and 202 do not need to be performed again, if the sorted object sequence has been stored when the procedure of ~~figure~~ Figure 2 was carried out the first time.

The resulting assignment of database tables to blades after the renewed performance of the procedure of ~~figure~~ Figure 2 with the lowered threshold T is shown in ~~figure~~ Figure 13. As apparent from the comparison of ~~figures~~ Figures 10 and 13 the load is more evenly balanced between the blades after the renewed assignment procedure.

Figure 14 shows an alternative approach for refining the object size balancing. In step 1400 Delta 2 is calculated by calculating the difference of the sum of the storage capacity of the blades and the sum of the object sizes of the objects to be assigned to the blades and by dividing the difference by the number of blades. In step 1402 the threshold is calculated by subtracting Delta 2 from the storage capacity. This threshold is the theoretical limit for the minimum storage capacities required on the individual blades in order to accommodate the objects if it where possible to distribute the objects with finest granularity.

In step 1404 the method of ~~figure~~ Figure 2 is performed again with the threshold as determined in step 1402 whereby the number N is fixed, i.e., for the last blade B_N which is processed. In one embodiment, the storage capacity will not be sufficient in most cases. In the resulting assignment of objects to blades, it is checked whether for the last

blade, which has been processed, there is in fact an excess amount of memory requirement, which exceeds the storage capacity.

If this is not the case, the assignment of objects to blades is outputted in step 1408. If the opposite is the case, the excess amount of memory is divided by the number of blades N which provides Delta 3. In step 1412 the threshold is incremented by Delta 3 and the control goes back to step 1404.

Steps 1404, 1406, 1410 and 1412 are carried out repeatedly until there is no longer an excess amount of memory.

Figure 15 is based on the example of ~~figure~~ Figure 10 and shows the threshold T as calculated in accordance with step 1402 of ~~figure~~ Figure 14. In the example considered here, the difference between the sum of the storage capacities of the blades and the sum of the table sizes is 3 GB. The 3 GB are evenly distributed over the 8 blades, which provides the threshold T.

If there is no excess amount of memory as a result of one iteration but a gap between the aggregated size of objects, which have been assigned to the last blade N, the procedure is continued in order to reduce the gap. This can be done by dividing the gap by the number of blades N and distributing the result over the blades by increasing the threshold correspondingly. The gap is calculated as follows: threshold T – sum of the sizes of the objects assigned to blade N.

In this instance the process is stopped if (i) there is no significant change from one iteration to the next (ii) the iterations toggle between different results, (iii) the standard deviation of the distribution of the objects does not improve or (iv) a maximum number of iterations has been reached.

Figure 16 shows the result of the assignment procedure of ~~figure~~ Figure 2, which has been performed with the threshold T as determined in step 1402, consistent with an embodiment of the present invention. As a result of the assignment procedure, there is an excess amount of memory E for blade B_8. In the example considered here, the excess memory amount E is 858 MB. In accordance with step 1410 the excess amount E

is divided by the number of blades N = 8. In accordance with step 1412 the resulting amount of memory Delta 3 = 107 MB is added to the threshold. Next the assignment method of ~~figure~~ Figure 2 is carried out again with the increased threshold, which provides the result as shown in ~~figure~~ Figure 17.

Figure 18 shows a further alternative for refinement of the object size balancing, consistent with an embodiment of the present invention. First the step 1400 of the method of ~~figure~~ Figure 14 is carried out in order to calculate Delta 2. Delta 2 is equivalent to the gap between the theoretical limit, i.e., the threshold as calculated in step 1402 of the method of ~~figure~~ Figure 14, and the storage capacity of a blade.

This gap is scanned by a stepwise variation of the threshold in order to identify an assignment of objects to blades which is balanced. The number of steps, i.e. the number of increments of the threshold, can be predefined or is user-selectable.

In step 1800 Delta 2 is divided by the number of increments, which provides Delta 4. In step 1802 the threshold is calculated by dividing the sum of the object sizes by the number of blades N. With this threshold the assignment method of ~~figure~~ Figure 2 is performed again in step 1804.

In step 1806 a statistical measure is calculated as a quality measure for the assignment of objects to blades obtained as a result of step 1804. For example, the standard deviation of the aggregated sizes of objects assigned to each one of the blades is calculated.

In other words, for each blade the total of the sizes of the objects, which have been assigned to the blade, is calculated. This provides one total size per blade. Next the standard deviation is calculated for the total sizes.

In step 1808 the threshold is incremented by Delta 4 and the control goes back to step 1804. This procedure is continued until the threshold has reached the storage capacity, i.e. the upper limit.

In step 1810 one of the assignments obtained as a result of step 1804 is selected on the basis of the overall statistical measure. For example, the assignment having the lowest standard deviation is selected.

Figure 19 illustrates this method with respect to the example shown in ~~figure~~ Figure 10, consistent with an embodiment of the present invention. The threshold T of 3712 MB is obtained by the calculation of step 1802. From there the threshold is stepwise increased in increments of Delta 4, which is Delta 2 = 384 MB divided ~~my~~ by the number of increments. For example, the number of increments is 100. For each assignment procedure the standard deviation of the table sizes assigned to blades is calculated for selection of one of the assignments. ~~Preferably the~~ The standard deviations are preferably calculated only for those assignments which fit onto the minimum number of blades.

Figure 20 shows a computer 108, which has processor 110 for running program 112. Program 112 may be a computer program product and has module 114 for sorting of objects by size and module 116 for assigning of objects to blades.

Further computer 108 has storage 118 for storing a table listing the objects and object sizes to be assigned to blades, storage 120 for storage of a storage capacity value of the blades and storage 122 for storing of the number of blades. Further computer 108 has interface 124 for coupling to workstation 126.

In operation, the table with the object names/numbers and object sizes is entered via interface 124 and stored in storage 118. This corresponds to the information shown in ~~figure~~ Figure 3.

Further a storage capacity value for the storage capacity of each individual blade is entered via interface 124 and stored in storage 120. In the example considered here, the storage capacity value is 4 GB.

~~Next program~~ Program 112 ~~is~~ may then be invoked. Program 112 sorts the table of storage 118 by size to provide a sequence of objects (cf. ~~figure~~ Figure 4). Next module 116 performs the method of ~~figure~~ Figure 2 in order to determine the minimum number

of required blades. This minimum number is stored in storage 122 and is outputted via user interface 124. This number can be a basis for a users investment decision for purchasing the number of blades to realize a data processing system being capable of handling the objects as listed in the table.

In addition, module 116 can perform the methods of ~~figure~~ Figure 11, ~~figure~~ Figure 14 and/or ~~figure~~ Figure 18 for refinement of the object size balancing.

Alternatively, computer 108 is one of the blades. In this instance computer 108 can dynamically change the assignment of objects to blades when the object size changes. This way frequent swapping operations for swapping objects between blades can be prevented.

~~List of Reference Numerals~~

~~100~~	~~cluster~~
~~102~~	~~processor~~
~~104~~	~~memory~~
~~106~~	~~network~~
~~108~~	~~computer~~
~~110~~	~~processor~~
~~112~~	~~program~~
~~114~~	~~module~~
~~116~~	~~module~~
~~118~~	~~storage~~
~~120~~	~~storage~~
~~122~~	~~storage~~
~~124~~	~~interface~~
~~126~~	~~workstation~~